EXPERIMENT - 26

Object Oriented Programming Lab

Aim

Write a program to read two numbers and then divide first no by second no and raise an exception if second number is zero.

Syeda Reeha Quasar
14114802719
4C7

EXPERIMENT – 26

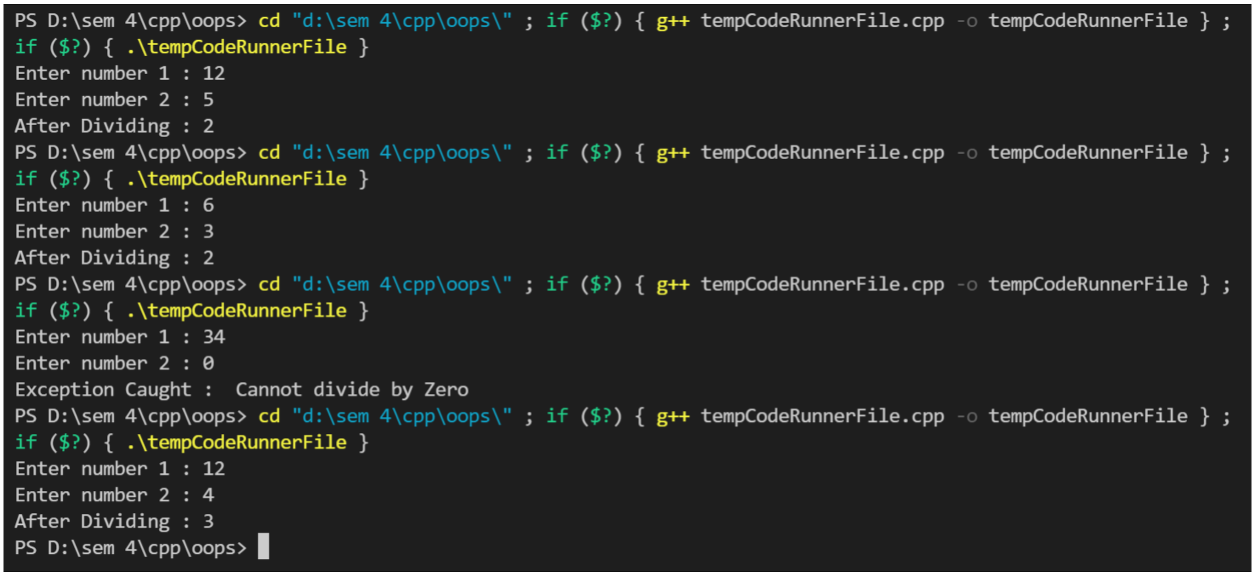
Aim:

Write a program to read two numbers and then divide first no by second no and raise an exception if second number is zero.

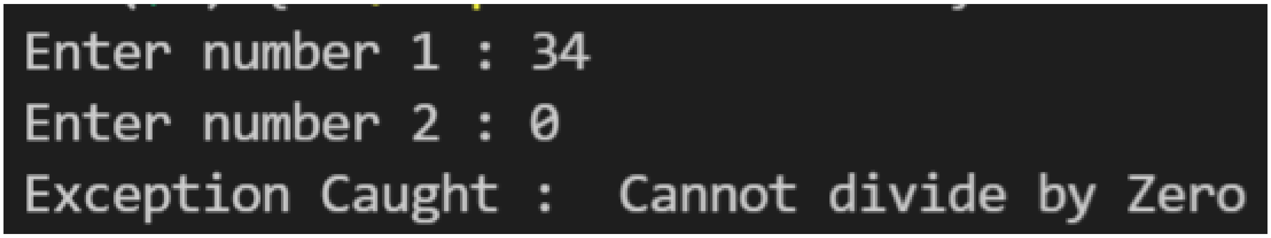
Source Code:

```
#include <iostream>

using namespace std;


int main()
{
    int x, y;
    cout << "Enter number 1 : ";
    cin >> x;
    cout << "Enter number 2 : ";
    cin >> y;
    try
    {
        if (y == 0)
        {
            throw y;
        }
        else
        {
            cout << "After Dividing : " << (x / y);
        }
    }
    catch (int)
    {
        cout << "Exception Caught :  Cannot divide by Zero";
    }
    return 0;
}
```

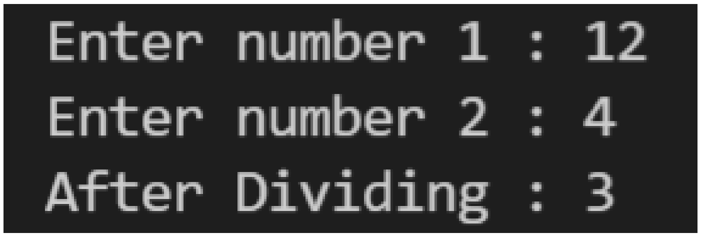
Output:



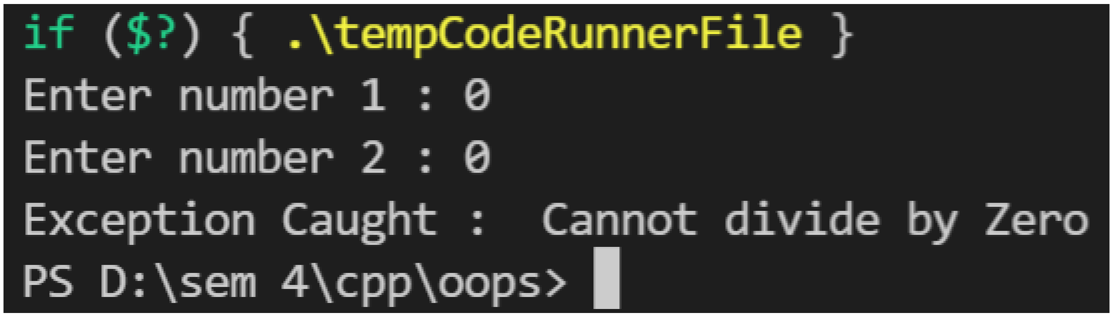
```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ tempCodeRunnerFile.cpp -o tempCodeRunnerFile } ;
if ($?) { .\tempCodeRunnerFile }
Enter number 1 : 12
Enter number 2 : 5
After Dividing : 2
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ tempCodeRunnerFile.cpp -o tempCodeRunnerFile } ;
if ($?) { .\tempCodeRunnerFile }
Enter number 1 : 6
Enter number 2 : 3
After Dividing : 2
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ tempCodeRunnerFile.cpp -o tempCodeRunnerFile } ;
if ($?) { .\tempCodeRunnerFile }
Enter number 1 : 34
Enter number 2 : 0
Exception Caught :  Cannot divide by Zero
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ tempCodeRunnerFile.cpp -o tempCodeRunnerFile } ;
if ($?) { .\tempCodeRunnerFile }
Enter number 1 : 12
Enter number 2 : 4
After Dividing : 3
PS D:\sem 4\cpp\oops> 
```



```
Enter number 1 : 34
Enter number 2 : 0
Exception Caught :  Cannot divide by Zero
```



```
Enter number 1 : 12
Enter number 2 : 4
After Dividing : 3
```



```
if ($?) { .\tempCodeRunnerFile }
Enter number 1 : 0
Enter number 2 : 0
Exception Caught :  Cannot divide by Zero
PS D:\sem 4\cpp\oops> 
```

Viva Questions

Q1). What is an exception or error handling?

Ans.

An exception is a problem that arises during the execution of a program. A C++ exception is a response to an exceptional circumstance that arises while a program is running, such as an attempt to divide by zero.

Exceptions provide a way to transfer control from one part of a program to another. C++ exception handling is built upon three keywords: **try, catch,** and **throw**.

- **throw** − A program throws an exception when a problem shows up. This is done using a **throw** keyword.
- **catch** − A program catches an exception with an exception handler at the place in a program where you want to handle the problem. The **catch** keyword indicates the catching of an exception.
- **try** − A **try** block identifies a block of code for which particular exceptions will be activated. It's followed by one or more catch blocks.

Assuming a block will raise an exception, a method catches an exception using a combination of the **try** and **catch** keywords. A try/catch block is placed around the code that might generate an exception.

Q2). What does throw exception mean?

Ans.

Exceptions can be thrown anywhere within a code block using **throw** statement. The operand of the throw statement determines a type for the exception and can be any expression and the type of the result of the expression determines the type of exception thrown.

Q3). What does catch exception mean?

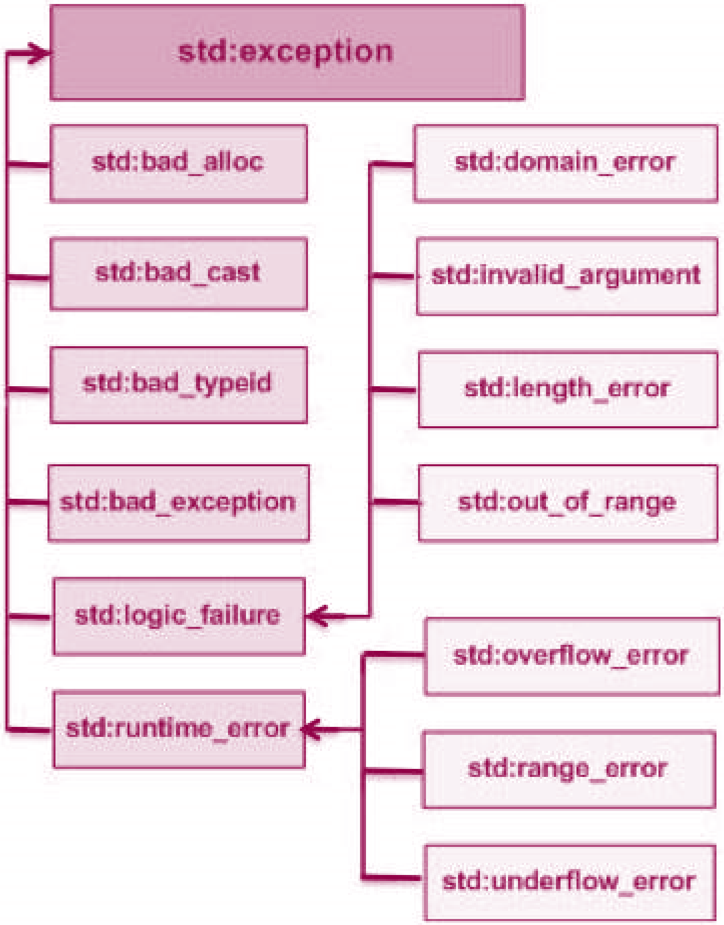
Ans.

The **catch** block following the **try** block catches any exception. You can specify what type of exception you want to catch and this is determined by the exception declaration that appears in parentheses following the keyword catch.

Q4). What are some standard exceptions in cpp?

Ans.

C++ provides a list of standard exceptions defined in **<exception>** which we can use in our programs. These are arranged in a parent-child class hierarchy shown below −


std:exception
std:bad_alloc
std:bad_cast
std:bad_typeid
std:bad_exception
std:logic_failure
std:runtime_error
std:domain_error
std:invalid_argument
std:length_error
std:out_of_range
std:overflow_error
std:range_error
std:underflow_error